



**This Page Blank (uspto)**

(19)日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11)特許出願公開番号

特開平9-212467

(43)公開日 平成9年(1997)8月15日

(51)Int.Cl. <sup>5</sup>	識別記号	庁内整理番号	F I	技術表示箇所
G 0 6 F 15/16	3 7 0		G 0 6 F 15/16	3 7 0 N
9/46	3 4 0		9/46	3 4 0 D
	3 5 0			3 5 0

審査請求 未請求 請求項の数3 O L (全 11 頁)

(21)出願番号 特願平8-13522

(22)出願日 平成8年(1996)1月30日

(71)出願人 000005223

富士通株式会社

神奈川県川崎市中原区上小田中4丁目1番  
1号

(72)発明者 伊藤 雅典

神奈川県川崎市中原区上小田中1015番地  
富士通株式会社内

(74)代理人 弁理士 岡田 守弘

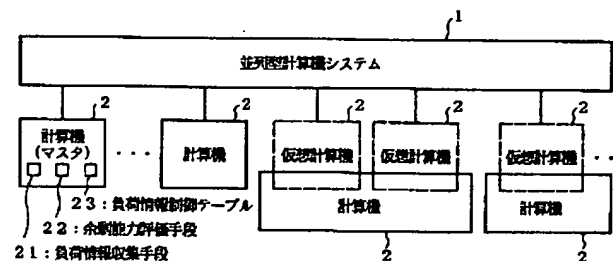
## (54)【発明の名称】 負荷分散制御システム

## (57)【要約】

【課題】 本発明は、負荷分散制御システムに関し、計算機ごとにCPU能力および台数が異なりしかも仮想計算機が混在する並列型計算機システムであってもパッチジョブを実行する計算機を自動的に選択してジョブを配送し負荷分散を図り、並列型計算機システムの処理効率を高めることを目的とする。

【解決手段】 各計算機の負荷情報の通知を受けて余剰能力を評価して算出する余剰能力評価手段と、いずれかの計算機にジョブが投入されたときに当該ジョブ情報の通知を受けて評価した余剰能力が最も高い計算機を選択し、ジョブを受け付けた計算機が最も余剰能力が高いときはその計算機にジョブを実行させ、一方、ジョブを受け付けた計算機以外の他の計算機が最も余剰能力が高いときはそのジョブを受け付けた計算機にジョブを転送させて実行させる手段とを並列型計算機の中のマスタ計算機に備えるように構成する。

本発明のシステム構成図



**【特許請求の範囲】**

【請求項1】並列型計算機の各計算機に負荷を分散する負荷分散制御システムにおいて、  
各計算機（仮想計算機を含む、以下同様）の負荷情報の通知を受けて余剰能力を評価して算出する余剰能力評価手段と、

いずれかの計算機にジョブが投入されたときに当該ジョブ情報の通知を受けて上記評価した余剰能力が最も高い計算機を選択し、ジョブを受け付けた計算機が最も余剰能力が高いときはその計算機にジョブを実行させ、一方、ジョブを受け付けた計算機以外の他の計算機が最も余剰能力が高いときはそのジョブを受け付けた計算機にジョブを転送させて実行させる手段とを並列型計算機のうちのマスタ計算機に備えたことを特徴とする負荷分散制御システム。

【請求項2】実記憶に対する負荷が所定負荷よりも高く過負荷と判明したときにその計算機を除外して他の計算機について余剰能力を評価して算出することを特徴とする請求項1記載の負荷分散制御システム。

【請求項3】上記余剰能力として、各計算機のCPU処理時間+CPU待ち時間、あるいはCPU処理時間+CPU待ち時間+I/O処理時間+I/O待ち時間としたことを特徴とする請求項1あるいは請求項2記載の負荷分散制御システム。

**【発明の詳細な説明】****【0001】**

【発明の属する技術分野】本発明は、並列型計算機の各計算機に負荷を分散する負荷分散制御システムに関するものである。

【0002】近年の計算機システムにおいて、業務処理容量の増大、処理速度の高速化、連続運用の必要性、信頼性の向上が要求されている。このため、仮想計算機を混在でき、システムの運用中に動的に稼働計算機の数を増減できる並列型計算機システムが提供されている。この並列型計算機システムでは、CPU能力の異なる計算機とCPU能力が可変の計算機が混在し、計算機によってCPU負荷状態が異なり、かつ並列計算機システムで稼働する計算機の数が動的に変動するため、並列型計算機システム全体の処理容量と処理速度を高めるためには各計算機のCPU負荷を平準化する必要がある。しかし、ユーザが各計算機のCPU負荷を平準化するのは困難であるために、システムが自動的にバッチジョブを実行する計算機を決定することが望まれている。

**【0003】**

【従来の技術】従来の並列型計算機システムは、図9に示すように、同じ能力のCPUを同じ台数備えた計算機を並列に接続する。この並列型計算機システムにおける負荷分散を行う手法は、OSが自計算機のCPU使用率を求めて他の全ての計算機に送信して知らせる。そし

使用率の低い計算機が余剰CPU能力が大きいとみなしてジョブを配送し、その計算機にジョブの実行をさせる（自計算機の余剰CPU能力が大きいたときは自計算機がジョブを実行する）。

**【0004】**

【発明が解決しようとする課題】上述した並列型計算機システムでは、並列型計算機システムを構成する計算機ごとに1台あたりのCPU能力が違う場合や、CPUごとの能力が同じでも台数が違う場合には、CPU使用率が同じでも余剰CPU能力が同じとは限らないから、CPU使用率の最も低い計算機にジョブを配送しても、ジョブを最も高速に実行できるとはならない問題が発生する。

【0005】また、並列型計算機システムには仮想計算機が混在できるが、仮想計算機には通常、次の3種類の動作モードがある。

・AUTOモード：同じ計算機上で走行する他の仮想計算機が要求するCPU能力と競合しない限り必要なだけ可変にCPU能力を使用できるモードである。

【0006】・上限AUTOモード：AUTOモードの仮想計算機と同じ計算機上で共存可能で、決められた比率（CPU配分比）しかCPU能力を使用できないモードである。

【0007】・ロジカルモード：AUTOモードや上限AUTOモードとは同じ計算機上で共存できないが、1台の計算機のCPU能力を任意の固定比率（CPU配分比）に分割して使用するモードである。

【0008】これらの仮想計算機システムはそれぞれ独立に、その上で走行するソフトウェアから認識できるCPU（論理CPU）の数を定義することができる。このため、上限AUTOモードとロジカルモードの仮想計算機は、実CPUの能力のうち配分比だけのCPU能力を持つ計算機と見なせるから、上記した問題が発生する。また、仮想計算機システムは、動的にCPU配分比を変更でき、上限AUTOモードとロジカルモードの仮想計算機において、異なる時刻に同じCPU使用率であっても、余剰CPU能力が同じとは限らないという問題も発生する。

【0009】また、計算機の実記憶負荷が高く、仮想計算機が過剰に動作し、外部ページと実ページで過剰に交換が行われている場合には、ページングI/O待ちが頻発してCPU使用率が低くなることがあり、この場合に、新たにジョブを動作させると、実記憶負荷が更に高くなり、ページングI/O待ちがより頻繁に発生し、CPU使用率が更に低くなって仮想計算機の処理効率が低下してしまう問題が発生する（この場合には、従来の負荷分散の手法では、ジョブを配送するのは逆効果となってしまう点で問題である）。

【0010】本発明は、これらの問題を解決するため、

計算機が混在する並列型計算機システムであってもバッチジョブを実行する計算機を自動的に選択してジョブを配送し負荷分散を図り、並列型計算機システムの処理効率を高めることを目的としている。

#### 【0011】

【課題を解決するための手段】図1を参照して課題を解決するための手段を説明する。図1において、並列型計算機システム1は、複数の計算機2および仮想計算機2を相互に接続して並列処理を行うためのものである。

【0012】計算機2および仮想計算機2は、ジョブを実行するものである。仮想計算機2は、計算機(実計算機)2に任意の個数を動的に設けることができる。ここで、計算機(マスタ)2は、負荷情報収集手段21、余剰能力評価手段22などから構成されるものである。

【0013】負荷情報収集手段21は、計算機(スレーブ)2から負荷情報などを収集するものである。余剰能力評価手段22は、各計算機から収集した負荷情報をもとに当該計算機の余剰能力を評価して算出するものである。

【0014】次に、動作を説明する。計算機(マスタ)2の負荷情報収集手段21が各計算機から負荷情報を収集し、余剰能力評価手段22がこの収集した負荷情報をもとに余剰能力を評価して算出し、いずれかの計算機2にジョブが投入されたときに当該ジョブ情報の通知を受けた計算機(マスタ)2が評価した余剰能力が最も高い計算機2を選択し、ジョブを受け付けた計算機2が最も余剰能力が高いときはその計算機2にジョブを実行させ、一方、ジョブを受け付けた計算機以外の他の計算機2が最も余剰能力の高いときはそのジョブを受け付けた計算機2にジョブを転送させ実行させるようにしている。

【0015】この際、実記憶に対する負荷が所定負荷よりも高く過負荷と判明したときにその計算機2を除外して他の計算機2について余剰能力を評価して算出し、最も余剰能力の高い計算機にジョブを実行させるようにしている。

【0016】また、余剰能力として、各計算機2のCPU処理時間+CPU待ち時間、あるいはCPU処理時間+CPU待ち時間+I/O処理時間+I/O待ち時間として算出するようにしている。

【0017】従って、計算機ごとにCPU能力および台数が異なり、しかも仮想計算機が混在する並列型計算機システムであってもバッチジョブを実行する計算機(仮想計算機を含む)2を自動的に選択してジョブを配送し負荷分散を図ることにより、並列型計算機システムの処理効率を高めることが可能となる。

#### 【0018】

【発明の実施の形態】次に、図1から図8を用いて本発明の実施の形態および動作を順次詳細に説明する。

図1において、負荷情報制御テーブル23は、計算機2から収集した負荷情報を記憶したり、これら記憶した負荷情報からジョブを実行させる計算機2を選択する時点で余剰能力としてエラップス期待値を計算して設定するものである(後述する図5参照)。ここで、エラップス期待値の最も小さい計算機2にジョブを配送するようにする。

【0020】次に、図2を用いて計算機(マスタ)2および図3を用いて計算機(スレーブ)2の構成を順次詳細に説明する。図2は、本発明の計算機(マスタ)例を示す。これは、図1の計算機(マスタ)2の詳細構成図であって、図1の負荷情報収集手段21は図2のCPU負荷情報収集部10に対応し、図1の余剰能力評価手段22は図2のジョブ配送先決定部9の一部に含まれるものである。

【0021】図2において、OS5は、オペレーティングシステムであって、全体を統括制御するものであり、ジョブ実行部7、ジョブ配送部8、ジョブ配送先決定部9、CPU負荷情報収集部10、CPU負荷情報受信部12、ジョブ配送先通知部13、ジョブ受付部14、ジョブ情報受信部16などから構成されるものである。

【0022】ジョブ配送部8は、ジョブ配送先決定部9によって決定された配送先にジョブを配送するものである。ジョブ配送先決定部9は、余剰能力の最も高い(エラップス値の最も小さい)計算機をジョブ配送先と決定するものである。

【0023】CPU負荷情報収集部10は、計算機2の負荷情報を収集するものである。CPU負荷情報受信部12は、スレーブ計算機2よりCPU負荷情報を受信するものである。

【0024】ジョブ配送先通知部13は、ジョブ配送先決定部9によって決定された配送先の計算機2にジョブを配送するようにスレーブ計算機に通知するものである。ジョブ受付部14は、投入されたジョブを受け付けるものである。

【0025】ジョブ情報受信部16は、スレーブ計算機に投入されたジョブの情報を受信するものである。図3は、本発明の計算機(スレーブ)例を示す。これは、図1の計算機(マスタ)2以外のスレーブの計算機2の詳細構成図である。

【0026】図3において、OS5は、オペレーティングシステムであって、全体を統括制御するものであり、ジョブ実行部7、ジョブ配送部8、CPU負荷情報収集部10、ジョブ受付部14、ジョブ配送先受信部16、ジョブ情報通知部17などから構成されるものである。7、8、10、14は図2と同一であるので説明を省略する。

【0027】図3において、CPU負荷情報通知部11は、CPU負荷情報などをマスタ計算機に通知するもの

マスタ計算機から受信するものである。

【0028】ジョブ情報通知部17は、投入されたジョブのジョブ情報をマスタ計算機へ通知するものである。以下図4ないし図8を用いて図1ないし図3の構成の動作を順次詳細に説明する。

【0029】図4は、本発明の動作説明図（その1）を示す。図4の（a）は、負荷情報収集のフローチャートを示す。図4の（a）において、ステージ1は、任意の計算機で実行するものである。

【0030】S1は、自計算機の負荷情報を収集する。この負荷情報は、例えば図4の（b）に示す①ないし⑨の情報を収集する。S2は、マスタ計算機に通知する。

【0031】図4の（a）において、ステージ2は、マスタ計算機で実行するものである。S3は、各計算機の負荷情報を受信する。S4は、負荷情報制御テーブルに格納する。

【0032】以上のステージ1のS1、S2およびステージ2のS3、S4によって、全ての計算機2の負荷情報およびI/O負荷情報を計算機（マスタ）2が収集し、後述する図5の負荷情報制御テーブル23のように設定（I/O負荷情報は未設定）できたこととなる。

【0033】図4の（b）は、ステージ1において収集・通知する負荷情報の例を示す。負荷情報は、図示の①ないし⑨の下記のものである。

①計算機識別子：並列型計算機システムを構成する各計算機を識別するもの

②CPU能力：実CPUの能力（実CPU一台当たりのMIPS値）

③CPU台数（1～N）

④実CPU台数（1～M）

⑤CPU使用率

⑥実CPU使用率

$$\begin{aligned} &= \text{CPU処理時間} + \text{CPU待ち時間} \\ &+ \text{I/O処理時間} + \text{I/O待ち時間} \end{aligned} \quad (\text{式1})$$

$$\begin{aligned} &= \text{CPU処理時間} \times (1 + \alpha (\text{CPU数}, \text{CPU使用率})) \\ &+ \text{I/O処理時間} \times (1 + \beta (\text{チャンネル数}, \text{チャンネル使用率})) \end{aligned} \quad (\text{式2})$$

$\alpha$ と $\beta$ は待ち行列理論の一般論から導かれるものである。この $\alpha$ と $\beta$ との関係は、例えばCPUバウンドなジョブであれば、必然的にCPU処理時間が大きくなり、I/O処理時間が小さくなるので、 $\alpha$ の大小関係に敏感に、 $\beta$ の大小関係に鈍感になり、CPU負荷情報およびI/O負荷情報をまとめて計算機2の余剰能力を評価してエラップス値として算出することが可能となった。

【0036】尚、I/O処理は、チャンネルと呼ばれる入出力機構を経由してディスク装置などと主記憶との間でデータの転送を行っている。1回のI/O処理にかかる時間は、チャンネル数やチャンネルの使用率に影響されるので、上記（式2）に示すようにCPUの場合と同様に評価するようにしている。

⑦CPU配分比：仮想計算機への実CPU能力の配分比（動的変更可能）

⑧計算機構成情報：実計算機、AUTOモード/上限AUTOモード/ロジックモードの仮想計算機の区別を表示

⑨実記憶負荷情報：スラッシングを起こしているか否かを表示

図4の（c）は、余剰能力の評価のフローチャートを示す。

【0034】図4の（c）において、ステージ1は、任意の計算機で実行するものである。S11は、ユーザが任意の計算機にジョブを投入する。S12は、ジョブが投入された計算機がマスタ計算機か、スレーブ計算機か判別する。マスタ計算機の場合には、ステージ2（図4の（a）のステージ2）に進む。一方、スレーブ計算機の場合には、S13でマスタ計算機に、投入されたジョブの情報を通知する。

【0035】以上のS11、S12によって、計算機に投入されたジョブ情報が全てマスタ計算機に通知されたこととなる。図5は、本発明の負荷情報制御テーブル例を示す。この負荷情報制御テーブル23は、既述した図4のS4で、全ての計算機2から収集された負荷情報を設定して記憶したものであって、既述した図4の（b）の①ないし⑨の情報を設定して記憶したものである。図中の“エラップス期待値”は、②ないし⑨の情報をもとに算出したものであって、計算機の余剰能力を表すものであり、小さいほど、計算機の余剰能力が高いものである。このエラップス期待値は、例えば下記の式によって計算する（尚、図5の負荷情報制御テーブル23は、CPU処理時間+CPU待ち時間についてのものである）。ジョブ投入からジョブ終了までに必要な時間（エラップス期待値）は、

示す。図6において、ステージ2は、マスタ計算機で実行するものである。S21は、任意の計算機より、投入されたジョブの情報を受信する。これは、スレーブ計算機に投入されたジョブの情報をマスタ計算機が受信、およびマスタ計算機に投入されたジョブの情報を受け付け、既述した図5の負荷情報制御テーブル23に設定する。

【0038】S22は、負荷情報制御テーブルを参照し、各計算機の負荷情報を順々に取り出す。S23は、実記憶負荷が過負荷か判別する。YESの場合には、S24ないしS26をスキップしてS27に進む。一方、NOの場合には、S24に進む。

【0039】S24は、計算機の種別を判別する。

は、S 2 6 に示す下記の式 4 によって当該計算機のエラー  
ップス期待値を評価し、S 2 7 に進む。

【0040】

$$(1 + \alpha (\text{実CPU数、実CPU使用率})) / \text{CPU能力} \quad (\text{式4})$$

・上限AUTOモード仮想計算機またはロジカルモード  
仮想計算機の場合には、S 2 5 に示す下記の式 5 によっ  
て当該計算機のエラーップス期待値を評価し、S 2 7 に進

む。

【0041】

$$(1 + \alpha (\text{実CPU数、実CPU使用率})) / (\text{CPU能力} \times \text{CPU配分比}) \quad (\text{式5})$$

S 2 7 は、負荷情報制御テーブルの最後まで評価を行っ  
たか判別する。YES の場合には、S 2 8 に進む。NO  
の場合には、S 2 2 に戻り、繰り返す。

【0042】S 2 8 は、エラーップス期待値が最小かつ実  
記憶過負荷でない計算機をジョブ配送先として選択す  
る。S 2 9 は、ジョブを受け付けた計算機に、選択した  
計算機をジョブ配送先として通知する。そして、図 7 の  
ステージ 3 へ進む。

【0043】以上の S 2 1 から S 2 9 によって、全ての  
計算機から受信した負荷情報を負荷情報制御テーブル 2  
3 に設定した後、先頭から順番に取り出して実記憶負荷  
が過負荷でない場合に計算機の種別によって分けてそれ  
ぞれエラーップス期待値を式 4 あるいは式 5 によって計算  
し、エラーップス期待値が最も小さい計算機にジョブを配  
送させて転送するように通知することが可能となる。そ  
して、後述する図 7 のステージ 3 によってジョブをエラ  
ップス期待値の最も小さい（余剰能力の最も高い）計算  
機に転送して実行させることが可能となる。

【0044】図 7 は、本発明の動作説明図（その 3）を  
示す。図 7 において、ステージ 3 は、ジョブを受け付  
けた計算機で実行するものである。

【0045】S 3 1 は、マスタ計算機よりジョブ配送先  
を受信し、ジョブをマスタ計算機から指示された計算機  
に配送する。ステージ 4 は、ジョブを配送された計算機  
で実行するものである。

【0046】S 4 1 は、ジョブを受け付けた計算機より  
配送されたジョブを実行する。以上の S 3 1、S 4 1 に  
よって、エラーップス期待値の最も小さい計算機にジョブ  
を配送して実行させることが可能となる。

【0047】図 8 は、本発明の余剰能力の評価例を示  
す。これは、CPU 負荷情報のみをもとに余剰能力を評  
価して計算したものである。以下説明する。ここで、あ  
るサービスを行う複数の窓口に対する客の到着頻度がラン  
ダム到着に従い、そのサービス量が指数分布に従うと  
仮定すると、到着した客がサービスを受けるまでの待ち  
時間は、処理時間  $\times$  （窓口の数と窓口の平均稼働率の関  
数）として図 8 の（式 6）として書けることが、待ち行  
列理論の一般論として知られている。今、窓口の稼働率

を CPU 使用率に、窓口の数を CPU 台数に対応させ、  
あるジョブを実行完了するために必要な時間を CPU 待  
ち時間と CPU 処理時間だけで評価すれば、CPU 処理  
時間は、（式 7）のように評価でき、ジョブの処理に必  
要なダイナミックステップ数はどの計算機で走行させて  
も同じだから除外して考えると、ジョブを投入してから  
終了するまでに要する時間に比例する量が得られる。こ  
の値（エラーップス期待値）が最も小さく、かつ実記憶負  
荷が過負荷状態でない計算機に対してジョブを配送し、  
実行させる。ここで、I/O 処理時間および I/O 待ち  
時間を既述した（式 1）、（式 2）のように含めるよう  
にしてもよい。

【0048】

【発明の効果】以上説明したように、本発明によれば、  
計算機ごとに CPU 能力および台数が異なり、しかも仮  
想計算機が混在する並列型計算機システムであってもバ  
ッチジョブを実行する計算機（仮想計算機を含む）2 を  
自動的に選択してジョブを配送し負荷分散を図る構成を  
採用しているため、仮想計算機を含む並列型計算機シ  
ステムのバッチジョブを実行させる際に、最も余剰能力の  
高い計算機を動的に選択してジョブを配送し実行させ、  
処理効率を高めることができる。

【図面の簡単な説明】

【図 1】本発明のシステム構成図である。

【図 2】本発明の計算機（マスタ）例である。

【図 3】本発明の計算機（スレーブ）例である。

【図 4】本発明の動作説明図（その 1）である。

【図 5】本発明の負荷制御情報テーブル例である。

【図 6】本発明の動作説明図（その 2）である。

【図 7】本発明の動作説明図（その 3）である。

【図 8】本発明の余剰能力の評価例である。

【図 9】従来技術の説明図である。

【符号の説明】

1：並列型計算機システム

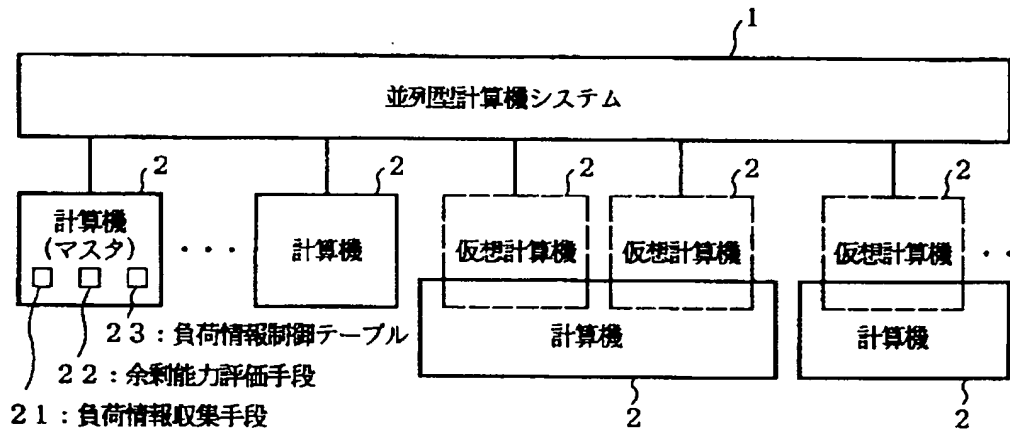
2：計算機、仮想計算機

21：負荷情報収集手段

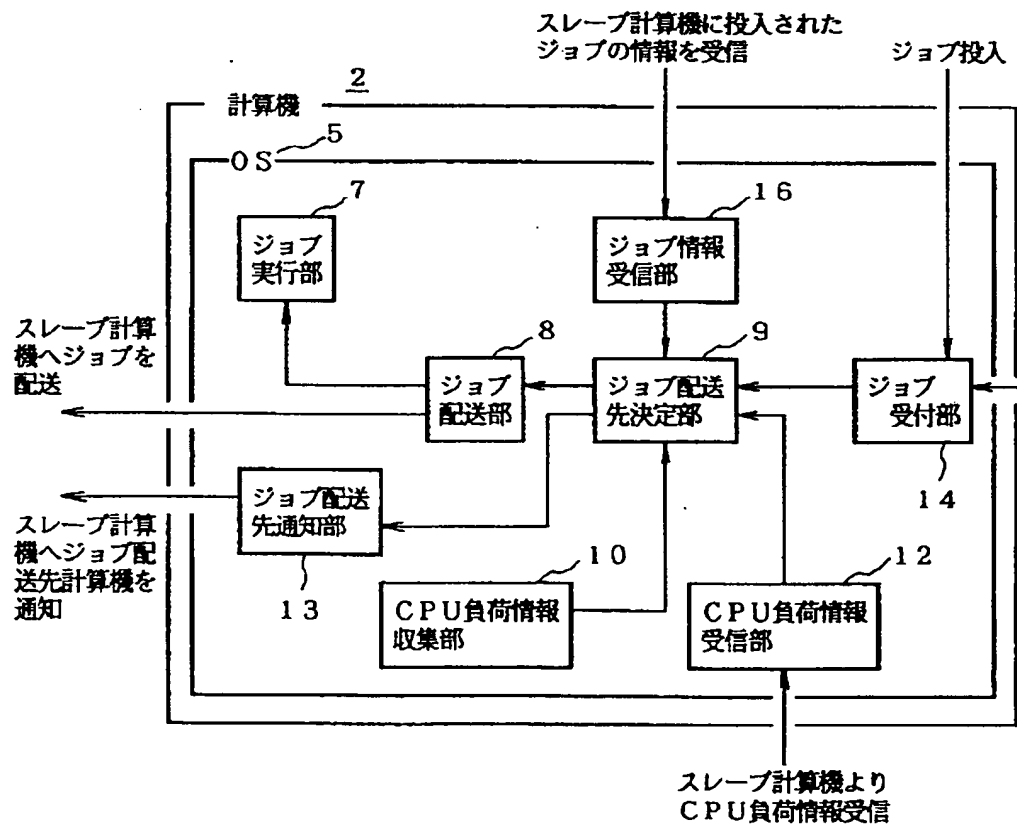
22：余剰能力評価手段

23：負荷情報制御テーブル

### 本発明のシステム構成図

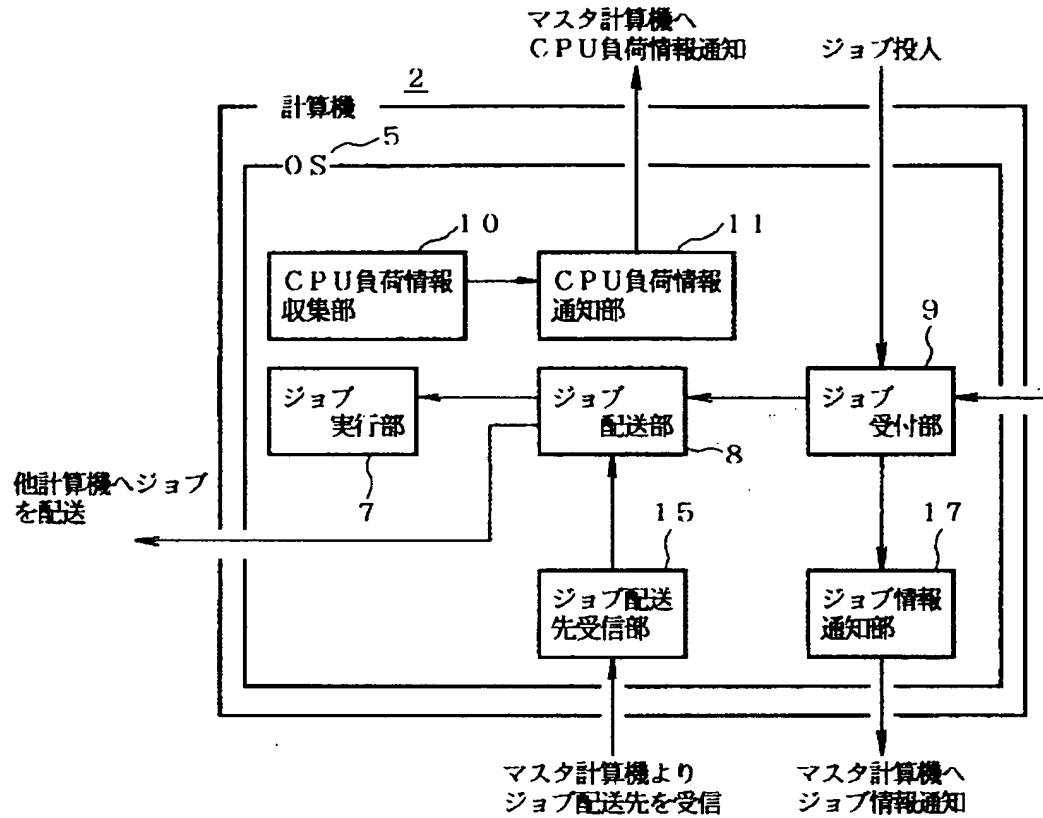


### 本発明の計算機（マスタ）例





### 本発明の計算機（スレーブ）例



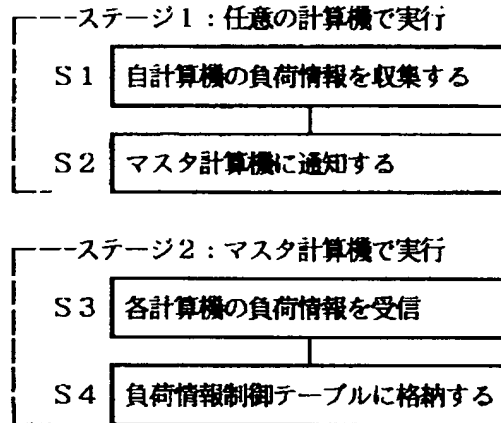
### 本発明の負荷情報制御テーブル例

[illegible]

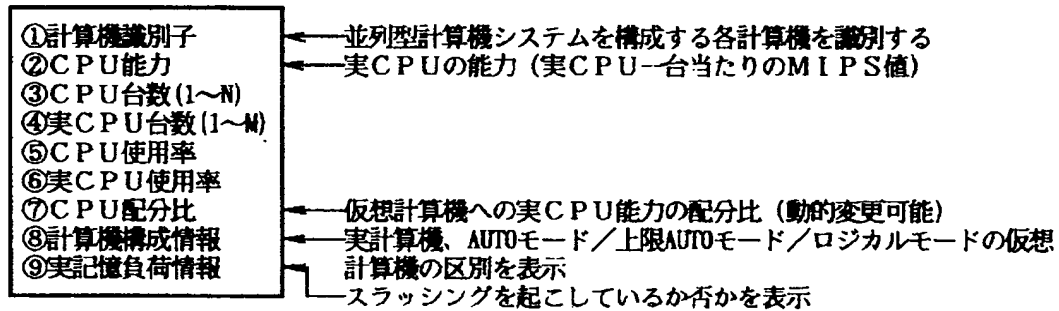
【図4】

## 本発明の動作説明図（その1）

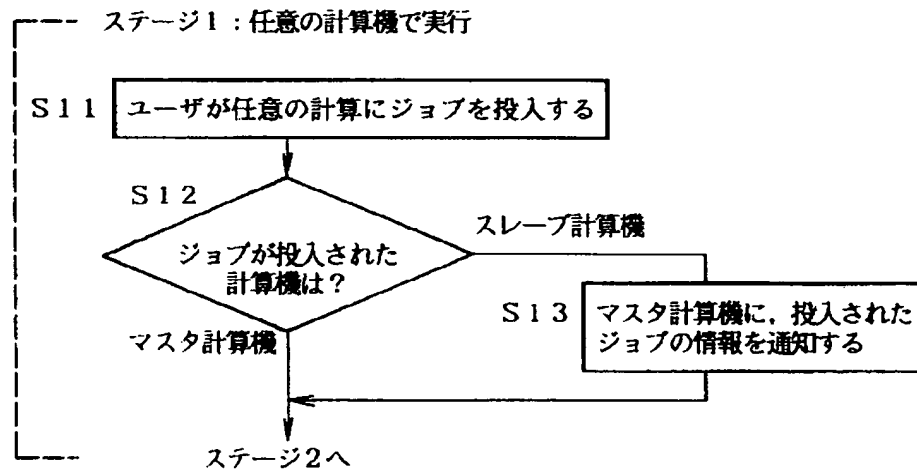
## (a) 負荷情報収集



## (b) ステージ1において収集・通知する負荷情報

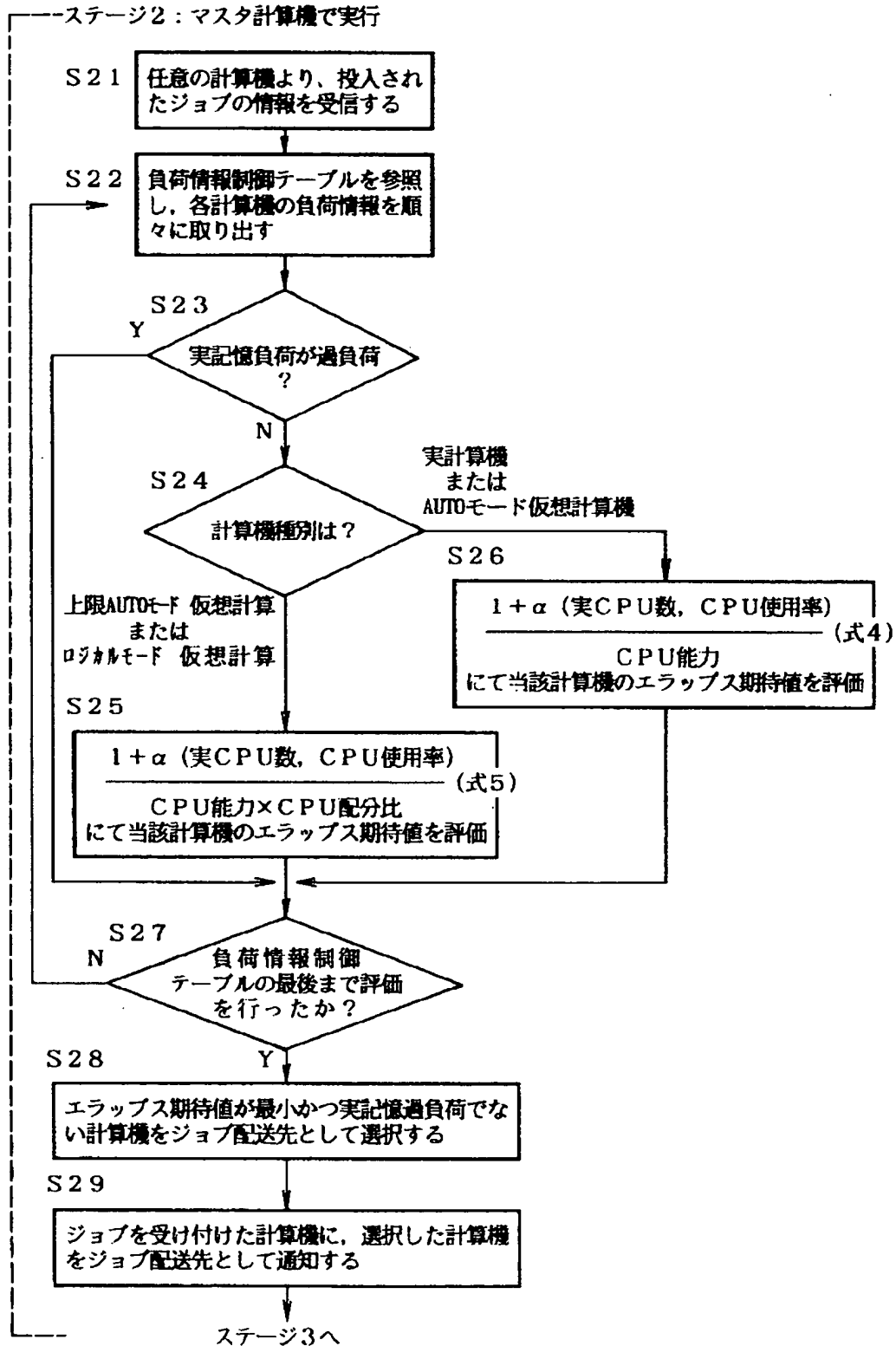


## (c) 余剰能力評価



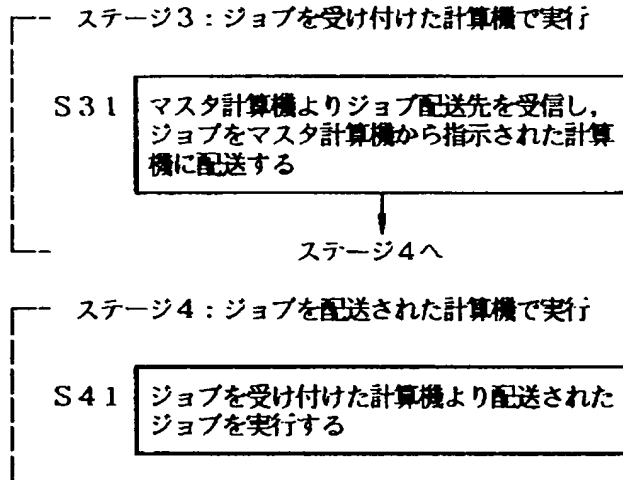
【図6】

## 本発明の動作説明図（その2）



【図7】

## 本発明の動作説明図（その3）



【図8】

## 本発明の余剰能力の評価例

(式6)

待ち時間 = 処理時間 × α (窓口の数 (s), 窓口の平均稼働率 (a))

$$= \text{処理時間} \times \frac{a^s}{(s-1)! \times (s-a)^2} \times \frac{1}{\sum_{n=0}^{s-1} \frac{a^n}{n!} - \frac{a^s}{(s-1)! \times (s-a)}}$$

$$0 \leq \text{窓口の平均稼働率 (a)} \leq 100$$

(式7)

CPU待ち時間+CPU処理時間

= CPU処理時間 × α + CPU処理時間

= CPU処理時間 × (1 + α)

ここでCPU処理時間は、

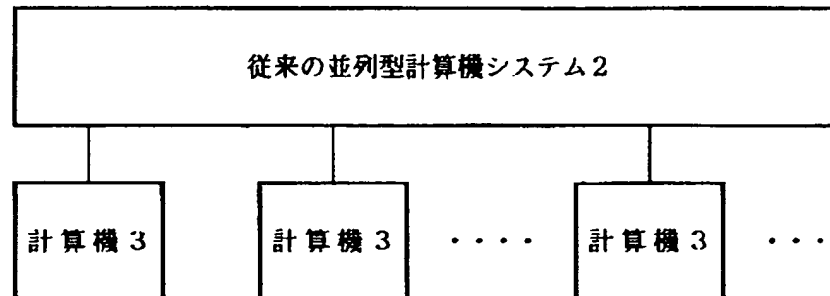
$$\text{CPU処理時間} = \frac{\text{ジョブを処理完了するのに必要なダイナミックステップ数}}{\text{CPU能力 (MIP値)}}$$

だから、最終的に (式4) を以下のように評価する

$$\text{CPU処理時間} + \text{CPU待ち時間} \propto \frac{(1 + \alpha (\text{CPU台数}, \text{CPU使用率}))}{\text{CPU能力}}$$

【図 9】

## 従来技術の説明図



**This Page Blank (uspto)**

**this Page Blank (uspto)**